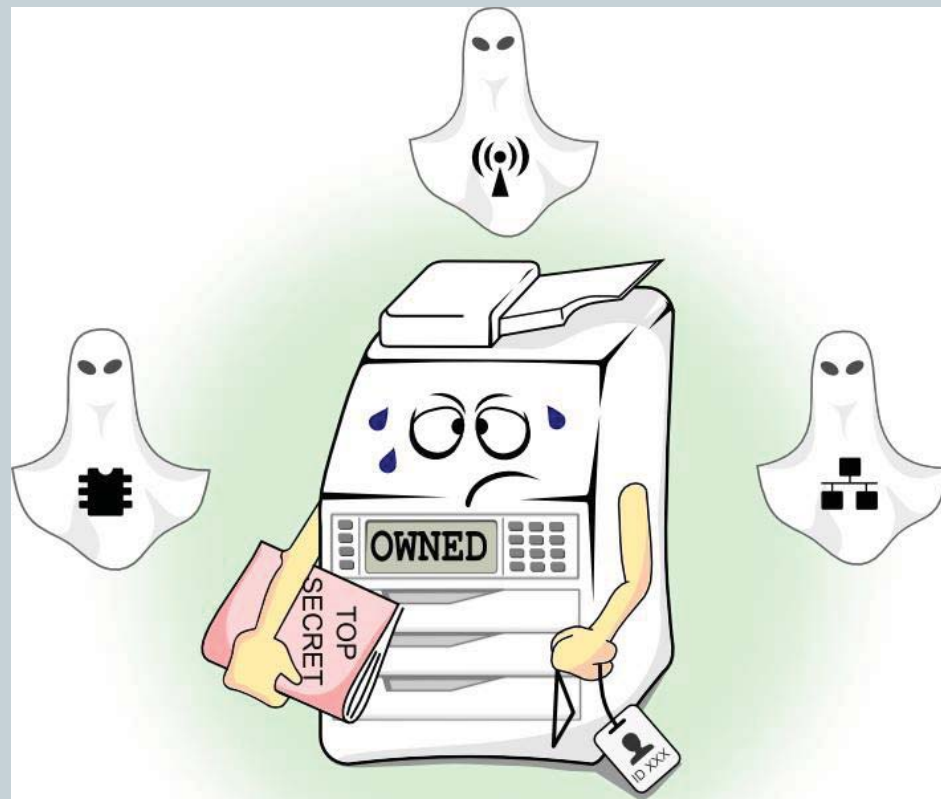


Hacking printers: for fun and profit



Impressum



- [Andrei Costin](#)
- Author of MFCUK
 - [MiFare Classic Universal toolKit](#)
- Day-time programmer (after-8pm type of hobbyist hacker)
 - Not part of printing industry though
- Generally interested in:
 - Programming/hacking: RFID, GSM, biometrics, embedded
 - Almost everything which:
 - ✦ Is connected to networks/communications lines
 - ✦ Have smart-cards (contact and contactless)
 - ✦ Have crypto involved somewhere down the line
 - ✦ Is or should be secure
 - Corporate/Enterprise IT support software & security
 - [TEMPEST](#) and [ISS](#)

Abstract



- While more and more new devices (routers, smartphones, etc.) are getting connected to our SOHO/enterprise environments, all-colour hats are getting plenty of focus on their security: defend and harden on one side; exploit and develop malware on the other.
- However, a special class of network devices (specifically network printers/scanners/MFPs), which are networked for more than 15 years, are constantly out of the modern security watchful eye.
- And even though we entrust them even the most confidential documents or the most sacred credentials (LDAP, PINs, RFID badges, etc.), we don't realize closely how weak and unsecured they are, despite the few minor security bulletins that started to pop-up here and there in the recent few months.
- In this presentation, we will try to analyze the reasons why hacking network printers/MFPs is a reasonable and accomplishable idea. Also, we will take a look at current state of (weak) affairs in the vulnerability and security research available. Then we will try to envision types of possible exploitation scenarios, backed-up with a printer remote-exploit demo. We will conclude the presentation with possible solutions and what can be done to protect ourselves as well as our network environments.

Disclaimer*



- No Warranties or Liability. Information is provided as-is, though every effort has been made to ensure the accuracy of the information presented. Author of the presentation is not legally liable under any circumstances for any damages such as but not limited to (including direct, indirect, incidental, special, consequential, exemplary or punitive damages) resulting from the use or application of the presented information.
- Unless explicitly noted in forms such as but not limited to "the XYZ Company says", etc., the opinions expressed in this presentation are solely and entirely my own. They should not be interpreted as representing the positions of any organization (past, present, future, existent, non-existent, public, private, or otherwise) with which I may or may not have been, are or are not, or will or will not be affiliated at some time in the past, present, or future.
- All trademarks and registered names are the property of their respective owners. All the effort has been made to link to the original material used as exhibition items in the presentation, and those items are property of their respective owners.
- This presentation: © 2010, Andrei Costin. Released under:



• *big fat one – because everybody *loves* fineprints

`\H1B%-12345X@PJL JOB` “HackingPrinters”



- This presentation is about:
 - Hacking “the PC inside printers/MFPs”
 - ✦ Why would someone hack a printer/MFP
 - ✦ How would someone hack “the PC inside printers/MFPs”?
 - ✦ How easy/feasible is MFP *firm*ware creation and exploitation
 - ✦ How to protect yourself and your so-much-loved MFP?
 - Laying foundation for further community security research/development/PoC
- This presentation is NOT about:
 - Printers’ display hack (RDYMSG, OPMSG, STMSG)
 - Printers’ embedded web-server hacks (mostly not)
 - Printers’ SNMP configuration hacks (mostly not)
 - Exhaustive guide to hack every and last MFP (not yet!)

MFPs Exploitation – Why?



- First, my term for MFP = Mfp, Fax, Printer
- Many would ask “Why would you exploit an MFP?” – answer derives from questions below:
 - How many persons would expect their MFP infected?
 - How many users/admins/security-auditors audit and hard-secure their/network MFPs?
 - ✦ Even if they do, do MFP vendor pay attention to security?
 - Bottom-line is always “It’s just a damn printer/MFP!”
 - How many persons or anti-malware products could clean such a malware?
 - ✦ Afaik, o(zero) antimalware products for (huge) printers/MFPs market
 - Why not (net/port/vuln)scan the network from a printer which is not suspected/cleanable?
 - Why not hide the malware/payload on a network printer and then make your way through the network/data?
 - Etc., etc., etc.

MFPs Exploitation – Why?



- First of all – (most) printers/MFPs are already full-blown computers! (or even space-ships ☺)
- Have goodies to play/own:
 - Some flavor of (RT)OS (VxWorks, LynxOS, Nucleus, Linux)
 - Embedded Java VM (eg.: ChaiServer)
 - Embedded Web Server (eg.: [Virata EmWeb](#))
 - Ethernet/WiFi
 - ✦ Not covering TCP/UDP/IP stack attacks, but there are [examples](#)
 - Eventually HDD – nice to scan/dump
 - ✦ Eg.: recent [CBSNews Investigation Case](#) – with much hype
 - Eventually SecureJet-like extensions – sweet thing ☺!
 - Eventually Fax board
 - Eventually Mailboxes

MFPs Exploitation – Why?

- MFPs interact with (hence can get access to):

- RFID badges



- Smart/swipe cards



- Fingerprints



- PINs



- LDAP/domain passwords

- Aren't these some-of **sweet** things we are hunting after all?

MFPs Exploitation – Why?



- Looking for confidential documents?
 - Why taking the trouble for infecting a PC-host on a network (eg. both elements being secured, updated & monitored) just to get a document with strong crypto using long-enough key and then not being able to decrypt it...
 - ...when instead wait for it to be in-printer decrypted (eg. SecureDimm) and printed (and I guess secret documents are still being printed on paper occasionally for selected eyes) so you get it decrypted in plain text

MFPs Exploitation – Why?



- Not so much information in this area (compared to PC or mobile devices)
 - [PJJ UPGRADE](#) – approx 6 results
 - [PJJ LPROGRAMENG](#) – 0 results
 - [PJJ LPROGRAMRIP](#) – 1 result (security paper)
 - [PJJ DMINFO](#) – approx 300 results
 - [PJJ DMCMD](#) – approx 75 results
 - Compare with this [PDF "/>](#)
- Too few known (more or less) public research:
 - slobotron, phenoelit, irongeek, Protek Research Lab's, DSecRG, SEC Consult + few other brave enthusiasts
- Recent disclosures mainly focused on web-admin, snmp, XSS and uncontrolled buffer overflows
 - Not too much detailed analysis on OS, kernel and firmware level

MFPs Exploitation – Why?



- Big number of devices – according to Gartner:

**Worldwide: Page Printer Vendor Shipment Estimates, 2005
(Thousands of Units)**

Company	2005 Shipments	2005 Market Share (%)	2004 Shipments	2004 Market Share (%)	2004-2005 Growth (%)
Hewlett-Packard	10,527,966	49.0	8,828,405	48.7	19.3
Samsung Electronics	1,874,820	8.7	1,901,933	10.5	-1.4
Lexmark	1,268,089	5.9	1,131,213	6.2	12.1
Brother	1,178,039	5.5	1,018,642	5.6	15.6
Canon	1,154,203	5.4	909,492	5.0	26.9
Other Vendors	5,468,926	25.5	4,322,420	24.0	26.5
Total	21,472,043	100.0	18,112,105	100.0	18.6

Source: Gartner Dataquest (February 2006)

- Theoretically, magnitude of 10 x mlns of devices (24 mlns/yr):
 - ✦ Perfectly exploitable & non-easy-cleanable
 - ✦ Always on, no antivirus & firewall running inside of them

MFPs Exploitation – Why?



- The Holy Grail would be to own “securities printers”
 - Currency/financial assets printing machines
 - ✦ Unfortunately limited to very closed circles ☹ - for obvious reasons
 - ✦ No updates/patches on internet to poke around
 - Industrial currency check/count machines
 - ✦ More or less accessible
 - ✦ From BPS 2000/3000 Banknote Processing Systems for Central Bank Applications “*The operating system software and all production data can be authenticated to protect data integrity and guard against tampering (optional)*” – isn’t it just sweet ☺
 - Passport/ID printing machines
 - Eg.: Oberthur, Giesecke&Devrient, others
 - These are not part of this presentation ☹... yet ☺!



Current main players



- Canon
- Fujitsu
- HP
- Konica Minolta
- Lexmark
 - Dell is selling Lexmark – [“So, Lexmark makes Dell's printers?”](#)
 - Eg.: BRQP205.fbb is for Lexmark E342N/Dell Personal Laser 1710
- Xerox
- Sharp
- Kyocera Mita
- Kodak
- Brother
- Samsung
- Toshiba
- Ricoh, Lanier, Nashuatec, Infotek, OCE, OKI

Current state of vulnerabilities



- [Xerox](#) – Total 44
 - [XRX04/10](#), [XRX05/9](#), [XRX06/7](#), [XRX07/2](#), [XRX08/10](#), [XRX09/4](#), [XRX10/2](#)
- HP – [CVE-HP-printer](#), [CVE-HP-MFP](#) = Total 20
 - [More](#) and [more](#)
- [Lexmark](#) – [CVE-Lexmark-printer](#) = Total 7
- Canon – [CVE-Canon-printer](#) = Total 2
- Kyocera – [CVE-Kyocera-printers](#) = Total 2
- OKI – [CVE-OKI](#) = Total 2
- Fuji – [CVE-Fuji](#) = Total 2
- Ricoh – [SBo5-005](#) = Total 1
- OCE – [CVE-OCE](#) = Total 1
- Brother – [CVE-Brother-printer](#) = Total 1
- Nashuatec – [CVE-Nashuatec](#) = Total 1
- Too few for such a complex, big & old industry!
 - This can't be true - the exploits are there waiting for us ☺

MFPs Exploitation – Real (miss)use scenarios



- PDOS aka bricking
 - Can be at most a teenage prank. Fun first 1-2 times.
 - HDMoore: *“It seems like if you can do a remote update of firmware, it would better to deliver a Trojan’ed firmware image, instead of just a DOS”*
- Idle-time processing
 - Port/network/exploits scanner
 - Computing/hash-cracking/sniffing
- Malware/upload storage
- “Stealth”/uncleanable command and control
- Unencrypted data theft
- Corporate/enterprise/intelligence assets data theft
 - Using SecureJet-like extensions
- Produce PDFs with o-day exploits
 - Just infect/replace the PDF output engine or replace output PDF file
 - Usually, DSS and scanners are trusted internal sources
- Spam inside/outside networks
 - Many devices have emailing capabilities (not all configured though)

MFPs Exploitation – Real (miss)use scenarios



- Ransomware (as it becomes more widespread)
 - Install the ransom-ware, which takes care to overtake the firmware upgrade module
 - So ransomware accepts only secured&signed upgrades/unlocks from it's creators – anything else rejected
 - Store & forward (if external connection detected) documents-to-print to the creator
 - But instead of printing any document, print something like:
 - ✦ “This printer is hijacked. Get unlock got from: www.printerhijacker.com using these details: [brand] [model] [serial_number] [ethernet_MAC] [other_bits]”
 - Based on printer model (it's price, year), the ransom amount can be decided (obviously a fraction of the catalog/second-hand cost)
 - If the victim pays, unlock code/firmware is provided (customized for that printer only based on serial#/MAC/etc)
 - Otherwise, victim risks to “loose” his/her device (sometimes quite expensive - \$32K)

MFPs Exploitation – *Futuristic/unreal* scenarios



- Espionage/blackmail (high-profile)
 - Very-unlikely, but possible. Target mainly models with HDD in high-profile organizations (those afford HDD models ☺)
 - Store the documents which hit keywords (Eg.: strategic, attack, intelligence, transactions, \$\$\$) – hint: good as MFPs AI research!
 - When storage is full, display [critical_dummy] error , document the error as “ship to 800-fake-service”, [get data from HDD](#), ship back ☺
 - [“Spies in the Xerox Machine”](#)
 - ✦ Russian saying goes *“Everything new is actually well-forgotten old”*
- Compete-ware
 - Want to make trouble to competitors? Increase their OPEX
 - Malware (for printer fuser especially) to run warm-up and high-power non-stop during office-nights
 - Eg.: HP LaserJet 4250dtnsl eats-up 750 watts – sweet bill

Main printer specifications



- Myriad of specs and languages... %)
- UEL – Universal Exit Language
 - Just one command
 - ✦ **<esc>%-12345X** (*<esc> is 0x1B aka \H1B aka ESCape*)
 - Harmless by itself, lethal in specific combinations
- PJL – Printer Job Language
 - Developed by HP
 - Job level controls: printer language switching, job separation, environment, status readback, device attendance and file system commands
 - Have essential security design flaws, hence exploitable
 - Examples:
 - ✦ **<esc>%-12345X@PJL JOB\r\n**
 - ✦ **@PJL RDYMSG = “Sample PJL Job”\r\n**
 - ✦ **@PJL ENTER LANGUAGE = PCL\r\n**
 - ✦ ...
 - ✦ **@PJL EOJ\r\n<esc>%-12345X**

Main printer specifications



- PCL – Printer Control Language
 - Developed by HP
 - Well, actually it's not a control language (PJT is)... name confusion...☺
 - It's more a formatting-control language, like PS
 - Harmless, but parsers and interpreters could be exploited
 - Examples:
 - ✦ Usually PCL jobs start with:
 - `<esc>E\r\n`
 - ✦ Sample commands in the job:
 - `<esc>&l1T`
 - Toggles the printer's job separation mechanism
 - `<esc>&l3X`
 - Instructs to print 3 copies
 - ✦ Mandatory PCL jobs end with:
 - `<esc>E\r\n`

Main printer specifications



- **PS** – PostScript Language

- Developed by Adobe
- Mostly formatting-control language, but has “device control” commands as well ☺
- On top, it is a programming language as well... (see later)
- Also, parsers and interpreters could be attacked
- Hence can be exploited
- Examples:
 - ✦ `%!PS-Adobe-3.0\r\n`
 - ✦ `%%LanguageLevel: 2`
 - ✦ `%%BeginFeature: *PageSize A4`
 - ✦ `1 1 sub ==`
 - ✦ `%%EOF\r`

- **PPD** – Adobe PS Printer Description

- Describe the entire set of [features](#) and capabilities available for their PostScript printers
- Contains the PostScript code (commands) – way to hack

Main printer specifications

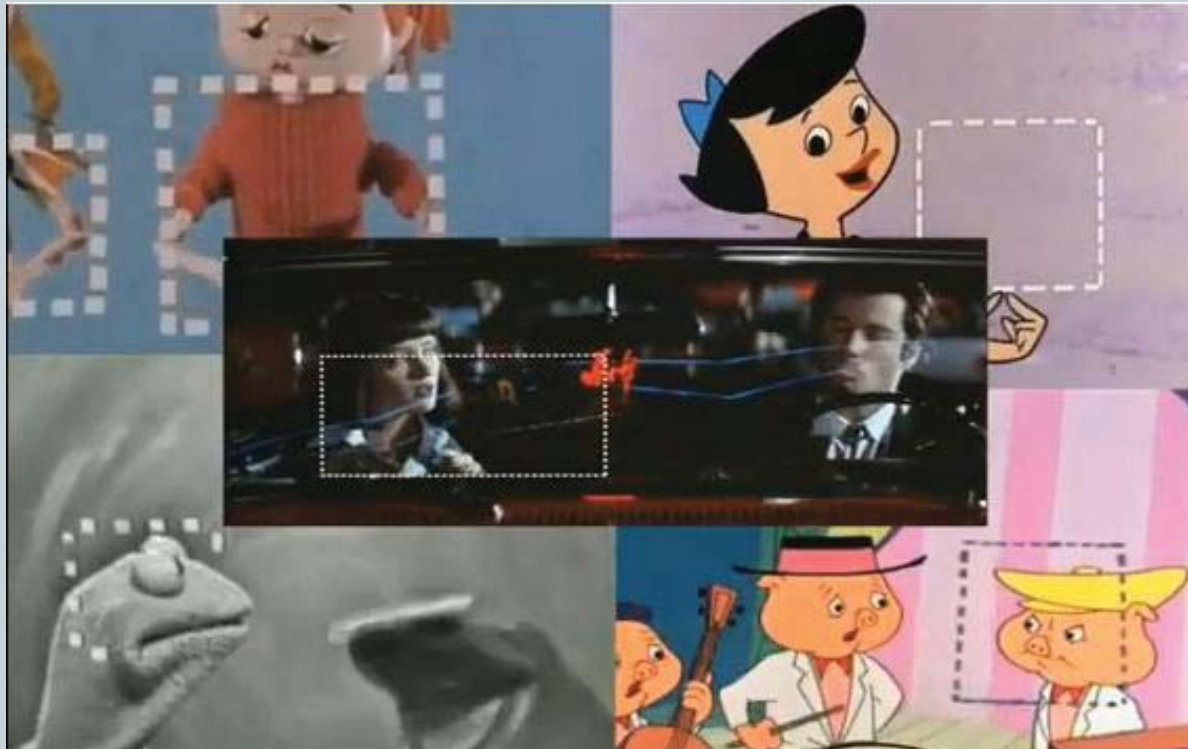


- PML – Printer Management Language
 - HP's object-oriented request-reply protocol to exchange device management information
 - PML can be used to query SNMP values from a printer device
 - So... turning SNMP off doesn't solve all problems ☺
 - Examples:
 - ✦ @PJL DMINFO ASCIIHEX="PmlRequest"\r\n
 - ASCIIHEX="PmlReply"\r\n\f
 - ✦ @PJL USTATUS TRAP\r\n
 - ASCIIHEX="PmlTrapRequest"\r\n\f
- GPD – Generic Printer Description
 - Windows GDI-based spec, similar to PPD
 - Used for creating unidrv.dll minidrivers for non-PS printers
 - Something like a customization plugin over unidrv.dll (not a bad idea)
 - Usually here: *c:\windows\system32\spool\drivers*
 - Examples of attack later

Specifications *“fineprints”



- PjL holes:
 - No provisions for standard, secure and vendor/arch/os-independent way for binary/firmware upload/upgrades
 - ✦ Everyone reinvented their own wheel – sadly, most did it wrong...



Specifications *“fineprints”



- PjL holes:
 - No standard provisions for strong authentication
 - No standard provisions for encryption
 - ✦ All usernames, PINs & passwords are in clear-text
 - @PJL SET USERNAME="HackingPrinters"
 - @PJL SET HOLDKEY="1234"
 - @PJL SET KMUSERKEY2 = "password"
 - Print job PIN security (@PJL HOLDKEY)
 - ✦ We are in 2010 – we get 0-9999 PIN/password range... ☺
 - ✦ Also, specs say nothing about N-tries-and-fails scenario actions
 - Again, the wheel...

Specifications *“fingerprints”



- PS/PPD holes:
 - setdevparams/setsystemparams
 - ✦ Can be powerful (and dangerous ☺)
 - ✦ Can be helpful, if you trust .PS file or know what you are doing
 - ✦ Can also set security/password settings on device – sweet
 - Think this: *.doc/*.pdf attacks PC, *.ps attacks MFP
 - Also, since PS is an interpreted programming language
 - ✦ Fuzz/smash the stack with [PS recursion](#) or stack operations
 - **Password* PS-field in the PPD file is in clear-text
 - PPD have nice **PatchFile* and **JobPatchFile* commands
 - ✦ Explained later

MFPs Exploitation – How to approach?



- Remote-initiated printing (RIP) exploiting channel
 - Java is our best friend here
 - Flash and Silverlight are not too friendly... yet
 - JavaScript is good as well – use [CrossSitePrinting](#)
 - Will [Google Cloud Printing](#) be as well? Time will show
- Locally-initiated printing (LIP) exploiting channel
 - MS Word can somewhat help us
 - Adobe LiveCycle XDC files can help us
 - GhostView is not too friendly yet

MFPs Exploitation – How to approach?



- Exploiting “test print” access in printers’ EWS
 - Not always available
 - Easy to patch – though easy patches are hard to get right for some...
- Internal interpreters’ exploit
 - PostScript, PCL – most widely used interpreters
 - Can borrow ideas from GhostScript exploits
- Locally-executed applications with rogue firmware
 - Requires social engineering
- Printer driver hacks
 - Requires either social engineering or admin-level escalation

Remote-initiated printing exploit

- Printing Payload Exploit (PPE) over Java Applets requires user intervention



- Lure the users to a site and then trick to print
 - Eg: print tickets, print discount coupons, print charity-related stuff, print government/tax related forms/discounts, etc.
- Can be successful using social engineering/nagging
 - Similar to [VBScript F1/Help Keypress Vulnerability](#)

- 



Demo – Remote-initiated printing exploit

- Printer exploited: reset, malware upload, etc.

The image is a composite screenshot illustrating a remote-initiated printing exploit. It features three main components:

- HP LaserJet 5200 Web Interface:** A browser window showing the HP LaserJet 5200 status page. The 'Device Status' tab is active, displaying 'Ready' and 'Supplies' information. The 'Print' button is visible at the bottom.
- Hacking Printer Web Interface:** A browser window showing a page titled 'Hacking Printer' with a large 'Print your ticket here' button. The page displays three 'South Africa 2010 FIFA World Cup' logos.
- Windows Command Prompt:** A terminal window showing the execution of a ping command to 10.27.2.20. The output shows multiple successful replies from 10.27.2.20, indicating the printer is online and responsive.

A large play button icon is overlaid on the center of the image, suggesting a video demonstration.

Remote-initiated printing exploit



- Possible exploitation problems
 - User doesn't check the box
 - ✦ This can be detectable by subsequent calls to java print services
 - ✦ Then annoy user until user checks the box (detectable by time-based analysis between java print services calls)
 - Printer name != precise target name
 - ✦ Java print services gives us only printer name ☹
 - ✦ Use 1 binary with all known printers exploits
 - Hope one sub-firmware hits the target, others will be discarded
 - Big data file is not quite invisible
 - ✦ Use “magic” detection (eg. like “%HP%”) and then fire one or a subset of firmwares

Locally-initiated printing exploit



- MS Word
 - “Print and get your printer owned” type of exploit
 - Will video demo in next slide
- Adobe [LiveCycle XDC files](#) (XML files)
 - Used in [SAP® environments](#)
 - “Infect”/replace all XDC files with required firmware payload
 - ✦ Doesn’t necessarily need admin rights
 - Good example how to do this is [here on page 15](#)

```
<xdp:xdp xmlns:xdp="http://ns.adobe.com/xdp/">
<xdc name="ps_plain" xmlns="http://www.xfa.org/schema/xdc/1.0/">
<pdl>
<seq id="preDoc"><ESC/>%-12345X@PJL RDYMSG DISPLAY=""&#13;&#10;
@PJL UPGRADE SIZE = 1024&#13;&#10;[hex encoded payload]<ESC/>%-12345X</seq>
</pdl>
</xdc>
</xdp:xdp>
```

Demo – Locally-initiated printing exploit

- “File upload” PPE over MS Word

Printing this page will upload a file to the printer's filesystem.

Select Telnet 10.27.2.20

```
@PJL PSDIRLIST NAME="0:" ENTRY=1
.. TYPE=DIR
.. TYPE=DIR
PostScript TYPE=DIR
PJL TYPE=DIR
saveDevice TYPE=DIR
WebServer TYPE=DIR
hackers.txt TYPE=FILE SIZE=36
```

Before LIP

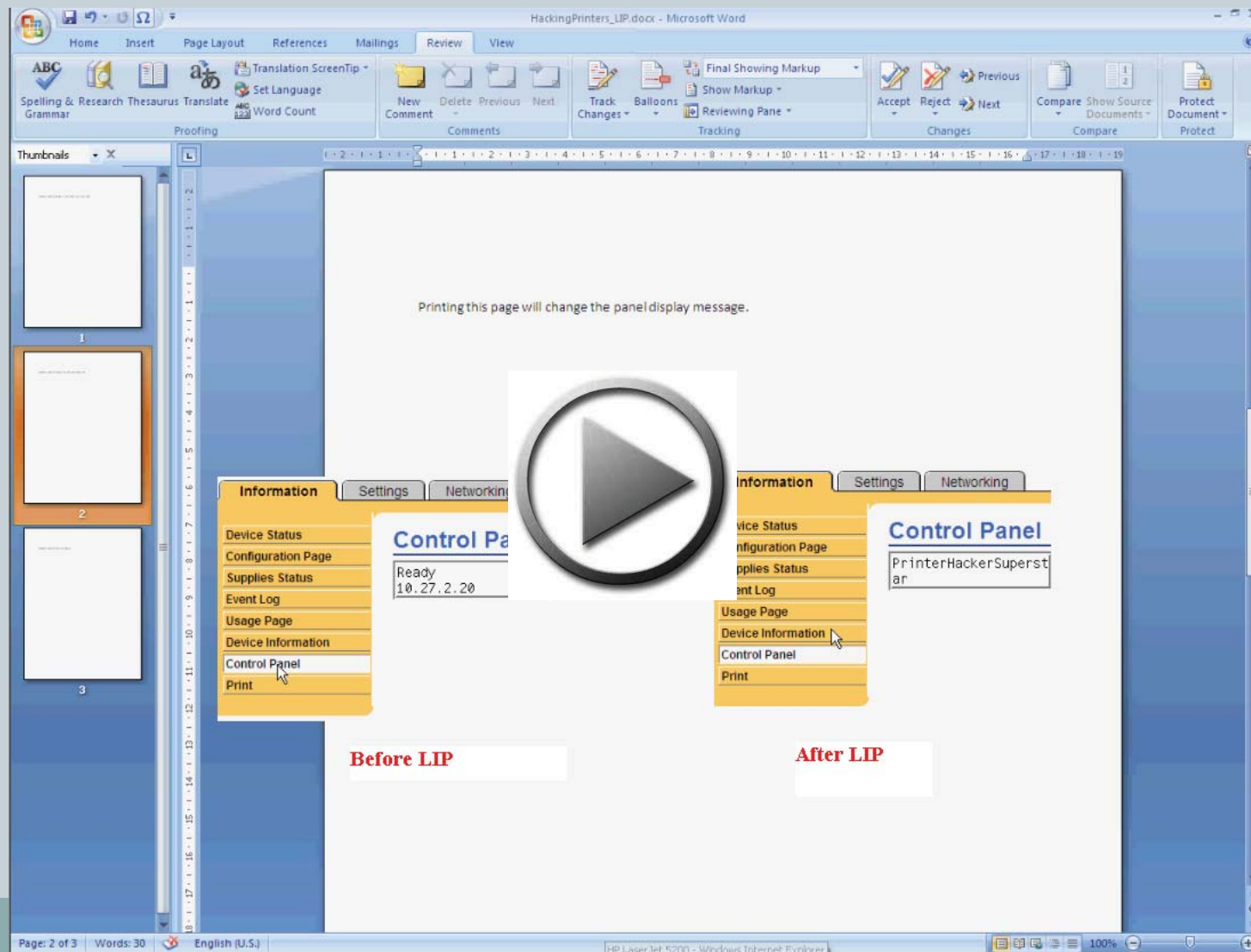
Telnet 10.27.2.20

```
@PJL FSUPLOAD FORMAT: BINARY NAME="0:\HackingPrinters.txt" OFFSET=0 SIZE=36
Your printer is hackers' superstar!
```

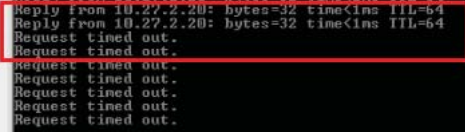
After LIP

Demo – Locally-initiated printing exploit

- “Printer-display change” PPE over MS Word



- 



Solutions for remote+local initiated exploits



- How to fix?
 - **Not easy**, since it's PjL design + device vendors' faults
 - Java, Word, LiveCycle, etc. have no big blame
 - ✦ They act as “channels” for delivering the exploits/malware/malicious commands
 - ✦ Rather than fixing channels, better fix specifications and devices
 - Perhaps correct PjL specs + follow standard and safe low-level communication with devices on top of PjL
 - Paranoid solution:
 - ✦ Print everything thru a virtual/proxy/[filtering printer](#)
 - ✦ That will filter out unsafe/suspect payloads (and alert!), producing “safe” docs to print on real devices
 - Unless the virtual printer has bugs/[is exploitable itself](#) 😊

Exploiting “test print” access in printers’ EWS

- Print is unprotected! (and leaks internal network IP)
 - Do vendors think diagnostics actions can be harmless?

hp LaserJet 4250 / 10.0.1.201

hp LaserJet 4250

Information [Log In](#)

Device Status
Configuration Page
Supplies Status
Event Log
Usage Page
Device Information
Control Panel
Print

Other Links
[hp instant support](#)
[Order Supplies](#)
[Product Support](#)

Print

Device Status **Sleep mode on**

Identify the document you want to print by using either option shown below, then select the Apply button

Note: To print 'print-ready' documents (e.g.: .ps, .pdf, .pcl, .txt), enter the document file name.

Option 1

Select the document to download from your hard disk or network file server.

Choose File [Browse...](#)

Option 2

Input the address of the document to access via the web. Type the address in a form such as:
http://www.(your_server).com/somefile.ps

Address

[Apply](#) [Cancel](#)

Exploiting “test print” access in printers’ EWS



- Accepts file as direct upload :
 - Filters based **only** on extension: txt, pdf, pcl, ps
 - Will **not** accept:
 - ✦ *print_my_hexor.rfu* or
 - ✦ *print_my_hexor.fmw*
 - Will accept:
 - ✦ *print_my_hexor.pcl!*
 - ✦ Yes, in PCL we can embed PJI UPGRADE/equivalent commands
 - Also, **extension check doesn't enforce content check**:
 - ✦ Rename *print_my_hexor.pcl* into *print_my_hexor.pdf*
 - And here we go again 😊
 - ✦ Example: use *HP_LJ5200_restart.pcl.pdf*

Exploiting “test print” access in printers’ EWS



- Accepts file as URL link to a printable document:
 - Exploit as in previous direct local upload
- Other interesting uses:
 - Check if printer can access external addresses (cool for command-and-control type of attacks)
 - Might reveal internal/external topology, as well as proxies along the way
 - ✦ If the chain is not properly configured and secured
 - Try to DoS the MFP in two types of [slowloris](#)
 - ✦ Attacker’s http-client “slowloris”es MFP’s EWS
 - ✦ Attacker’s http-server “slowloris”es the MFP’s initiated http-clients to our URL-document
 - ✦ Do both from above simultaneously ☺
 - Find race conditions in parsers: direct print, direct URL print, port 9100 print and print-server print; include also PjL/non-[PDL](#) cmds

PostScript interpreters exploitation



- PostScript interpreters have bugs as well
 - GhostScript exploitable on your PC
 - “MfpPsInterpreter” exploitable on your MFP
- Stack and recursion are nice weapons
 - %[% Error: **execstackoverflow**; OffendingCommand: --nostringval--]%%
 - This is simple, but more complex/inconsistent stack operations can be done
 - ✦ Fuzzing the interpreter and stack is a good way to find out

```
[%-12345X@PJL JOB NAME="PostScript StackSmash"
@PJL SET STRINGCODESET=UTF8
@PJL SET HOLD=OFF
@PJL SET RESOLUTION=600
@PJL SET BITSPERPIXEL=1
@PJL ENTER LANGUAGE=POSTSCRIPT
%!PS-Adobe-3.0
%%Title: PostScript StackSmash

/stacksmash_loop {
    dup -1 sub
    stacksmash_loop
    mul
} def

1 stacksmash_loop ==

%%EOF
%%-12345X@PJL EOJ
%%-12345X
```


PostScript interpreters exploitation



- PostScript-related exploits
 - [CVE-2004-1717](#) – [Remote](#) buffer [overflow](#)
 - [CVE-2007-6725](#)
 - [CVE-2008-0411](#) / [CESA-2008-001](#) – Stack-based buffer overflow
 - [CVE-2008-6679](#)
 - [CVE-2009-0196](#)
 - [CVE-2009-0583](#) / [CVE-2009-0584](#)
 - [CVE-2009-0792](#)
 - [CVE-2009-4195](#) – [Buffer overflow](#)
 - [CVE-2009-4270](#)
 - [CVE-2009-4897](#)
 - [CVE-2010-1628](#)
 - [CVE-2010-1869](#) – [Stack](#) based [overflow](#)
- Try/tweak them out on your MFPs fleet
 - Some might surprise you

Locally-executed apps with rogue firmware



- If all other fail
 - Because of: fixes in webserver, script-blockers, etc.
- Social engineer the user to “download and play a nice game” application
- Doesn’t have to be a PC virus, a valid app will do ok:
 - It will be just a printer malware
 - So zero antivirus detection guaranteed still 😊
- Just connect to TCP port 9100 printer job spooler
- Dump the exploit/malware
 - Use @PJJL UPGRADE style commands
 - Use @PJJL FS* style commands

Locally-executed – Various findings



- TCP port integer overflow ([corporate-style](#))
 - Port 9100 is actually: $0 < 9100 + k * 0x10000 < 999999$
 - ✦ Hence: $k \in [0..15]$ – will all print OK to 9100 ☺
 - Not found yet a practical exploitation use

Configure Standard TCP/IP Port Monitor

Port Settings

Port Name: IP_127.0.0.1

Printer Name or IP Address: 0300.168.000.0x00008D

Protocol

☒ Raw ☐ LPR

Raw Settings

Port Number: 992140

LPR Settings

Queue Name:

☐ LPR Byte Counting Enabled

☐ SNMP Status Enabled

Community Name: public

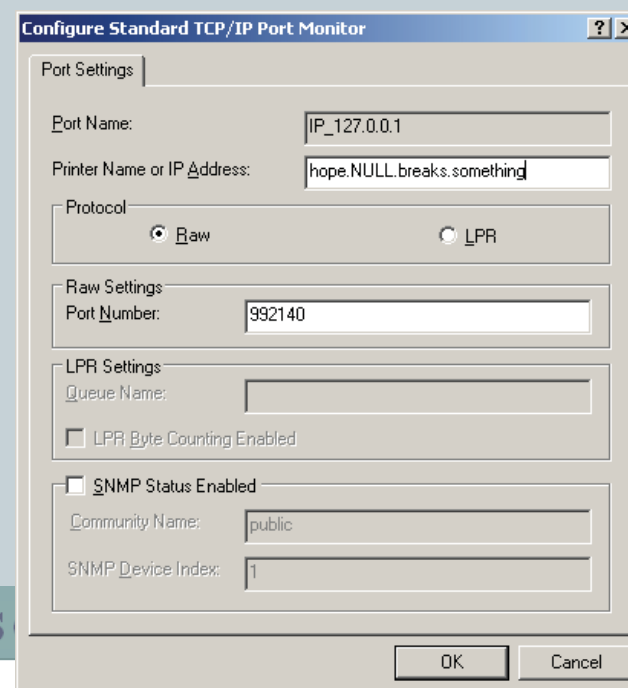
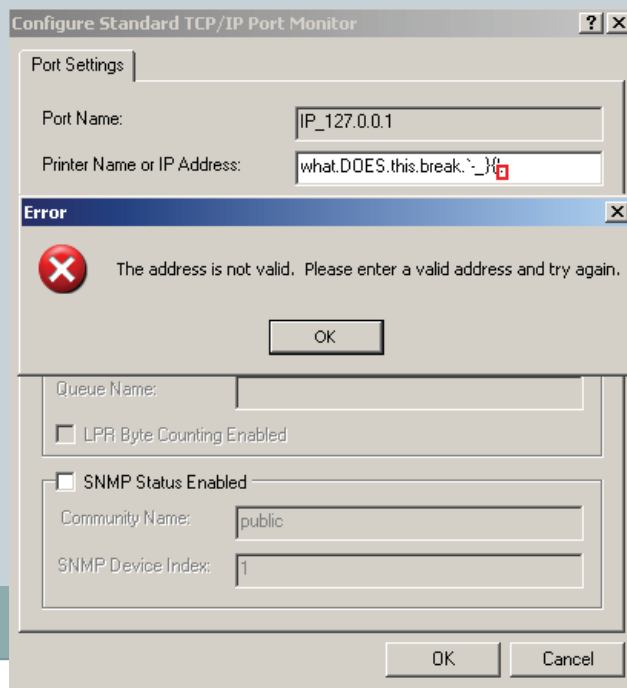
SNMP Device Index: 1

OK Cancel

Locally-executed – Various findings



- IPv4 validation dilemma (it's 2010 dudes! wtf?)
 - Accepts: dec, hex, oct ☺ - not necessarily new/scary
 - ✦ Eg.: 0300.168.000.0x00008D = 192.168.0.141
 - Cares last char not to be “.” (dot) – everything else is “just fine”
 - Seem to try to resolve anything else as hostname2ip (255.255.255.255 [.255], this.IP.is.INVALID)



Locally-executed – Printer driver hacks



- Find exploit stream for `unidrv.dll`/[pscript5.dll](#)
 - Possibly get LOCAL SYSTEM privileges ([spoolsv.exe](#))
 - `unidrv/pscript5` dlls called from user space, no need for admin
- Other require social engineering+admin level
 - Replace the driver *.dlls
 - Provide an “enhanced” driver, with printer-malware inside
- “Infect” the GPD files
 - Replace with legitimate *Cmd containing malware payload

```
*Command: CmdSelect
{
    *Order: DOC_SETUP.4
    *Cmd: "<1B>§§-12345X@PJL SET RESOLUTION=150<0A>@PJL ENTER LANGUAGE=PCL<0A0D1B>E<1B>*t150R"
}

*Command: CmdSelect
{
    *Order: DOC_SETUP.4
    *Cmd: "<1B>§§-12345X@PJL ENTER LANGUAGE=UPGRADE<0A0D1B><DE><AD><BE><EF><13><37>"
}
```

Locally-executed – Printer driver hacks



- “Infect” the PPD files
- **PatchFile*, **JobPatchFile*
 - Represents a PS language sequence that is a downloadable patch to ROM code or into initial VM
- **FileSystem*
 - The **?FileSystem* query can be used to dynamically determine whether or not a file system is actually present

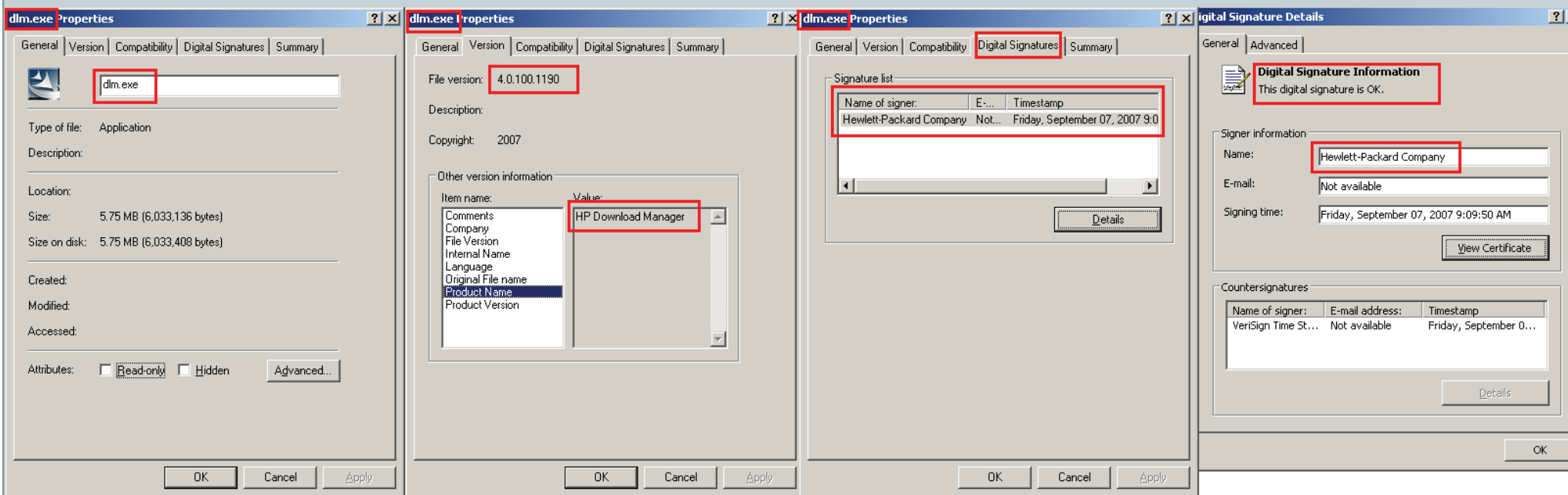
```
*JobPatchFile 1: "  
%  
%%BeginResource: LH PatchFile  
%  
false(  
800344A251010CD0613719CCA731D08205041C8DC6031190C04032170C61A728  
...
```

```
*?FileSystem: "  
  save false  
    (%disk?%)  
    { currentdevparams dup /Writeable known  
      { /Writeable get {pop true} if } {pop} ifelse  
    } 10 string /IODevice resourceforall  
  restore"  
*End
```

Privacy/transparency concerns



- Not satisfied with [printer tracking dots](#)?
- Satisfaction guaranteed with:
 - [HP Download Manager](#) – a story from backstagedoor
 - Will present minimal analysis of *hpjdwlnld.exe*



Privacy/transparency concerns

○ Important note:

- ✦ It's not managing a PC-backdoor
- ✦ It is managing an MFP-backdoor
- ✦ *strings* utility is enough to spot it
- ✦ Checks for %HOME%\upgrades\jetdirect\SpecialUpgrades.txt
- ✦ Checks special firmware files for ShortStack/CodeImage microcodes
- ✦ If you have samples for above 2 items, please share them!
- ✦ Possibly similar to [AMD K8 Microcode backdoor update feature](#)

○ Have *few others* HP call-home features under investigation

○ Are vendors being responsible when including backdoor/call-home features?

- ✦ Well-known PR fiascos: [Energizer](#), [Sony](#)

Address	Length	Type	String
"..."rdata:0045...	0000002A	C	FirmwareFileManager::CreateDiscDictionary
"..."rdata:0045...	00000030	C	FirmwareFileManager::GetFirmwareImageFileHeader
"..."rdata:0045...	00000033	C	FirmwareFileManager::IsMicroCodePartitionAvailable
"..."rdata:0045...	00000026	C	FirmwareFileManager::ReadBackDoorfile
"..."rdata:0045...	0000002E	C	FirmwareFileManager::ReadFirmwareBackDoorFile
"..."rdata:0045...	00000031	C	FirmwareFileManager::ReadFirmwareImageHeaderFile
"..."rdata:0045...	00000027	C	FirmwareFileManager::getCurrentVersion

DevEnv – How?



- My vision (yours might be slightly/totally different)
 - Unpack/mount the firmware
 - ✦ Need to reverse most important formats out of myriad
 - ✦ Crack any crypto + signature is a “desirable option” of course
 - Map it's arch + OS
 - ✦ Wiki, hex-view, specs, IDA, obj* suite
 - Fine-tune IDA, binutils, obj* army for that specific combination
 - Reverse the workings of (each) specific executable
 - Introduce the payload:
 - ✦ Byte-patch, if you talk code-machine better than your native lang
 - ✦ Compile a binary in an emulated env (if all prerequisites permit)
 - Test payload:
 - ✦ Directly on hardware – tricky, may brick it, need **good** HW skills, etc.
 - ✦ In an emulated env – very convinient, but again not always possible

DevEnv – Why?



- A DevEnv+Emulator tandem is preferred for:
 - Vendor firmware testing for vulnerabilities (parsers, etc.)
 - Develop malicious payloads/firmware for a device/device-class
 - Allows easier fuzzing
 - Is a more formal approach, rather than trial-and-error
- Unless:
 - You want a BIG net of BIG bricks (not bots) and BIG angry corps on your 455!
 - You own a [warehouse](#) of MFPs for tests

DevEnv – What?



- Toolchains:
 - [Crosstool-ng](#), [buildroot](#), [scratchbox](#)
- [Emulators](#)
 - [Qemu](#), [OVP](#), [RTEMS](#), [ARMuLator](#)
- OSes on most printers/MFPs:
 - [LynxOS](#)
 - [VxWorks](#)
 - NucleusOS
 - Linux (for various non-std architectures)
 - pSOS
- Processors on most printers/MFPs:
 - MIPS (PCM-Sierra)
 - RISCs (Toshiba TMPR4955)
 - ARM (Marvell ARMv5TE-compliant, custom HP-ARM)
 - SPARC (Fujitsu MB86830 series)

DevEnv – first things first – Linux



- Lexmark luckily went [Linux/GPL](#)
 - But VxWorks and LynxOS are not out-of community potential/knowledge
- Best start for devenv setup & research bootstrap
 - *E23x_E33x_141_C20.FLI* is a good kernel-loading example
 - Interacts with NVRAM and other stuff (good to understand)
 - Have “|*BIN*” wrapped image of Linux kernel
 - ✦ Can also be built from sources, though EAN.KA.K009 not released

```
0001fc00h: 00 00 00 00 00 00 00 01 00 00 00 00 58 A7 7C 42 ; .....XS|B
0001fc10h: 49 4E 00 1A BE FE FF AA 00 D7 FF AA 00 CO FF AA ; IN..%py².xÿ².Àÿ²
0001fc20h: 00 C1 00 00 00 64 FF AA 00 C5 62 65 61 66 01 00 ; .Á...dÿ².Àbeaf..
0001fc30h: 01 00 00 00 CO 00 00 06 3C CA 4B 65 72 6E 65 6C ; ....À...<Kernel
0001fc40h: 00 00 00 00 00 00 00 00 00 00 45 41 4E 2E 4B 41 ; .....EAN.KA
0001fc50h: 2E 4B 30 30 39 00 00 00 00 00 30 00 00 00 00 00 ; .K009.....0.....
0001fc60h: 00 00 00 00 1B BE 00 00 50 00 00 00 00 00 00 00 ; .....%...P.....
0001fc70h: 00 00 00 00 00 02 62 6C 64 2D 6C 69 62 00 00 00 ; .....bld-lib...
0001fc80h: 00 00 00 00 00 00 57 65 64 20 4E 6F 76 20 32 39 ; .....Wed Nov 29
0001fc90h: 20 31 35 3A 31 30 3A 35 30 20 32 30 30 36 00 00 ; 15:10:50 2006..
```

DevEnv – Firmware Unpack/Mount



- Firmware image unpackers:
 - Simple script-like C-tools
 - Do not work yet with encrypted firmware package
 - Strip proprietary-PJL wrappers and spit binary raw inside
 - ✦ Some have a single ELF file (example: E23X_.fli)
 - ✦ Some have a FS-like object with tree-structure and binary content
 - Can adopt and use [libPJL](#) from phenoelit
- Ultimate goal:
 - File-based FS drivers
 - To be as simple as:
 - ✦ **./mount -t hp-fru HPLJ5200.fru /mount/fw_test**

DevEnv – Firmware Unpack/Mount



- Example: excerpt from a single block of HP simple-FS, many of these found inside a single RFU firmware file:

```
00000600h: 00 00 00 04 00 0C 00 01 2E 00 00 00 00 00 00 03 ; .....
00000610h: 00 0C 00 02 2E 2E 00 00 00 00 00 05 00 0C 00 03 ; .....
00000620h: 6C 69 62 00 00 00 00 06 00 14 00 09 77 65 62 53 ; lib.....webS
00000630h: 65 72 76 65 72 00 00 00 00 00 07 00 10 00 06 ; erver.....
00000640h: 69 6D 61 67 65 73 00 00 00 00 00 08 01 B8 00 04 ; images.....
00000650h: 64 61 74 61 00 00 00 00 00 00 00 00 00 00 00 00 ; data.....

#define PJL_UPGRADE          "@PJL UPGRADE SIZE="

// small endian architecture
/*
 4 bytes ID/parent of entry,
 2 bytes size of entry (ENTRY_SIZE),
 2 bytes length of entry_name (ENTRY_NAME_LEN),
 ENTRY_NAME_LEN bytes having name,
 (ENTRY_SIZE - 4 - 2 - 2 - ENTRY_NAME_LEN) bytes
*/
typedef struct _fsentry_fix_
{
    unsigned char entry_id[4]; // 4 bytes
    unsigned char entry_size[2]; // 2 bytes
} fsentry_fix;

typedef struct _fsentry_var_
{
    unsigned char entry_name_len[2]; // 2 bytes
    char *entry_name; // variable
} fsentry_var;

typedef struct _fsentry_
{
    fsentry_fix fix;
    fsentry_var var;
} fsentry;
```

Sample firmware under microscope



- BarSTORM barcode printers
- Linux FS image with default unsalted passwords
 - root:\$1\$\$I2o9Z7NcvQAKp7wyCTliao:10933:0:99999:7:::
 - lp:\$1\$\$RfHkehRv/LWAGZdCEvUU90:10933:0:99999:7:::
 - bcadmin:\$1\$\$YSpLiaVeoDkQidsOLxlm5/:10933:0:99999:7:::
 - engineer:\$1\$\$YSpLiaVeoDkQidsOLxlm5/:10933:0:99999:7:::
 - admin:\$1\$\$I2o9Z7NcvQAKp7wyCTliao:10933:0:99999:7:::
 - crypt("password")=\$1\$\$I2o9Z7NcvQAKp7wyCTliao

Sample firmware under microscope



- IBM [InfoPrint IP 6700](#)
 - 369676.prg
 - [pRiNtrOnIx](#) firmware and components
 - ✦ Seems like a H4XoR designed the firmwares ☺
- Has RFID from [awidasia](#)
 - SDK and samples to play with are [here](#)
- Some keywords to get you interested:
 - PaRtITiOn OF RFID, rfidfirm.bin, rfidchip.inf, rfidtag.inf
 - AWID MPR-1510 V2.6h UHF MODULE Firmware Ver4.27
 - Why not spy on RFID tags or [KIL](#) all tags?

```
Silicon,TID,EPC Word Length,USR Word Length,TID Word Length,ACS/KIL supported?
```

```
Impinj Monza 1,E200104.,6,0,2,No
Impinj Monza 1a,E200105.,6,0,2,Yes
Impinj Monza ID,E200106.|E200108.,6,0,2,No
Impinj Monaco 64,E200106.|E200108.,6,4,2,No
Impinj Monza 2,E200107.,6,0,2,Yes
Impinj Monza 3,E200109.,6,0,2,Yes
Alien Higgs 2,E2003411,6,0,2,Yes
Alien Higgs 3,E2003412,6,0,6,Yes
Phillips Ucode EPC Gen2,E2006001,6,14,2,Yes
Philips Ucode G2XL,E2006004,15,0,4,Yes
Philips Ucode G2XM,E2006003,15,32,4,Yes
ST Micro XRAG2,E2007240,16,0,4,Yes
```

Sample firmware under microscope



- SMS Printers (examples)
 - Eg.: DPSPPro, Gatetel, Possio GRETA, Secugis
- Empty paper roll DOS attack (most printers)
 - Avg ~ 62 SMSes (160 new-line chars each SMS) for 50m rolls
- Configuration commands attack
 - Against [DPSPPro](#). Others might have hidden conf commands as well!
 - “#V1: 0=SMS 1=VOICE CALL [0]. This variable chooses whether the Printer will confirm with an SMS or placing a call.”
 - “#Y: Programs ACCEPT number to which ACCEPT SMS will be sent. Note1: if the CALL option is enabled, the unit will place a call instead.”
 - Make it call your/friend's premium number is the answer ☺
- Nice to have – reflash by TPDU-SMS ☺

“Secure Thinking” *in quotes*



- HP Security Solutions

- “Q23. Are current HP multifunction printers susceptible to viruses and worms? **No, since the majority of viruses and worms exploit vulnerabilities in Windows-based computers.** HP MFPs use non-standard operating systems other than Windows. Consequently, they are immune to these viruses and worms. **In practice, there have been no known instances of viruses or worms infecting HP MFPs**”
 - ✦ Well, PoC-community or some haxor or some IT-criminals might change that “in practice” then!
- “Firmware generally behind software in terms of secure development & deployment” – more than true
 - ✦ Wonder if HP's SecLab [PhlashDance](#) ever reached HP's MFP R&D

“Secure Thinking” *in quotes*



- Sharp Security Suite

- *“Sharp MFP products use unique embedded firmware and are not based on Windows operating systems. Therefore, **Sharp MFP’s internal systems are not subject to the same Virus vulnerability as Microsoft operating systems.** We believe this approach provides the internal systems of our products with protection against common Windows executable viruses and other similar infectious software programs.”*

- ✦ Well, possibly are vulnerable to *other* (i.e. not same) virus vulnerabilities!

“Secure Thinking” *in quotes*



- [Lexmark MFP Security](#), [Samsung MFP Security](#)
 - “In other areas, the security considerations around printers/MFPs are substantially different: ***they generally don’t run conventional operating systems, they don’t have network file shares that need to be secured, they probably don’t need or support antivirus software, etc.***”
 - ✦ Who did copy from who that text? Or they just assumed *the leader* is right and mutually-copy-pasted?
 - ✦ “...probably...” ?!
 - Nowadays, if you have an OS, a FS and externally connected execution environment, most likely you need internal antivirus/IDS/IPS

“Secure Thinking” *in quotes*



- Final thought on above “secure thinking” quotes
- Remember [psybot](#)? To summarize
 - Non-conventional arch – *true* – MIPS
 - Non-conventional OS – *true* - Mipsel Linux
 - Doesn't support antivirus – *true* – “why should we?!”
 - Got owned – *very true* – ~100k devices in a sophisticated command-and-control botnet
- If you need more arguments for securing/cleaning embedded devices, running unconventional OS+arch which do not support secure/antimalware standards/frameworks
 - ✦ Perhaps security is your *lowest priority hobby* – my \$0.02...

Solutions – Printer Vendors' Side



- First, accept that present day printers (especially network ones) are:
 - Full-blown computers themselves
 - A security target/threat
 - To be considered as part of Secure Development/Testing/Audit Lifecycles
- Fix those specs and parsers (PJL, PCL, PML, PDF, PS)
- Fix those damn web/telnet/ftp/snmp/etc. interfaces
- If first random 200 bytes fuzz string [crashes/bricks your device](#)...
 - ...time to put in practice [SDL](#). we are in 2010, remember?...

Solutions – Printer Vendors' Side



- Authenticate uploader, crypt, sign and verify signature of the uploaded firmware
 - Btw, homebrew or kindergarten crypto is NOT crypto!
 - Or make (some) implementations FOSS – so open and secure standards can be implemented (oh, these utopist ideas...)
- Be fair!
 - Transparent and backdoor-free systems/software
- Collaborate with antimalware vendors for your platforms
 - Could win you a nice marketing step
- Last but not least – remove default passwords and make mandatory strong-password changes as part of the initial setup procedures/installations

Solutions – Antimalware Vendors' Side



- Collaborate with vendors and security community
 - Make vendors understand those MFPs are real exploitable targets
 - Also, it could be a good marketing step “First antimalware on printers/MFPs”
 - Develop open and secure practices/protocols for in-printer antivirus management and updates
- If above collaboration does not work
 - Sponsor high-profile MFP exploit botnet – volunteers are out there
 - You have your foot in the “MFP antimalware market”`s door
 - This point is more to be joke ☺
 - Though, not that there were no surprising developments
- Setup honey-pots for most-spread MFPs EWS :
 - Similar to renowned /etc/passwd
 - Study blackhats/bots actions to train IDS/IPS for MFPs
 - Get samples of firmware or exploit payloads (PJL, PS, PCL)
- ... even though AV concept is being considered obsolete

Solutions – Admins' Side



- Develop and **follow** secure periodic practices and checklists for all your MFPs/printers
- Use and analyze extensive logging using MFPs management platforms
- Properly isolate MFPs on appropriate network segments
- Perhaps implement stricter domain-level printing policies
- Well, last but not least – don't leave those default accounts/passwords on

Solutions – Users' Side



- Stay updated to latest firmware of the printer's vendor
 - Make sure you choose a security-aware vendor (but skip the marketing BS between the lines)
- Don't print anything from untrusted sources
 - Well, this is hard... everybody is untrusted today
- Don't open unknown files
 - Not guaranteed that malware detection is triggered for printers-related malware
 - Important point – exploits the MFP, no need for admin rights on PC!
- Log and monitor printers' activity
 - Connects from it's IP
 - Paranoid mode – USB data filter from the printer to host PC
 - ✦ You never know what bugs do printer's driver have on the PC
- Use safe virtual printers to produce malware-free docs

Conclusions



- As PoC shown, printers are exploitable
- Specs have holes and are outdated for the new IT security realities:
 - Device and antimalware vendors seem to ignore the issues... yet
- MFPs are more than “dummy printers” – these are full-blown machines with great power and connectivity
- MFPs tend to interact with same (or even bigger) number of technologies as computers:
 - Eth
 - WiFi
 - RFID
- MFPs have access to almost same set of secrets as PCs

Credits/Props/Recommended reading



- [Slobotron on Hacking Printers](#)
- [phenoelit's HP resources](#)
- [Irongeek's "Hacking Network Printers"](#)
- [SANS Auditing and Securing Multifunction/MFP Devices](#)
 - Amuzing note: "Using this port and the right utility you can, among other things, *change what shows up on the LCD display*. Modification of the LCD panel, either causing confusion ("Out of Service") or opening the door for social engineering purposes ("Error. Call 555-5151.").

Credits/Props/Recommended reading



- [“Vulnerabilities in Not-So-Embedded Systems”](#)
- [“Exploiting Printers by Analyzing Their Firmware”](#)
(nowhere to find on the net... censored?!)
- [“Juste une imprimante”](#)
- [“Network Printing”](#) book
- [MFP Security for Enterprise Environments](#)
- [cyrtech.de](#)

@PJJL COMMENT = “Insert coin to continue”



● ?



● !



\H1B%-12345X@PJJL EOJ “HackingPrinters”



- Print-in-touch:
 - `lpr -Phoneypot-printer@andreicostin.com -Y -J “Hacking Printers” -T “Comments/suggestions/collaboration” -m andrei@andreicostin.com -m zveriu@gmail.com -- -`
- Till next time... keep your MFPs safe as golden:

